IN THE SPECIFICATION

Please amend the paragraph from page 3, line 30, to page 4, line 13, as follows:

--The principles of the preferred embodiments described herein have general applicability to the implementation of K-line wavelet filters. However, for ease of explanation, the embodiments ~~has~~ have been described with reference to a K-line 5-3 wavelet filter for use in a forward discrete wavelet transform of digital images. However, it will be readily evident that the invention is not limited thereto. For instance, the wavelet filter can also be used in K-line 9-7 wavelet filters. Furthermore, it would be apparent to a person skilled in the art[[,]] that the wavelet filter can be modified for use in an inverse discrete wavelet transform. Still further, it would be apparent to a person skilled in the art that two such wavelet filters can be combined in known manner to provide a 2-dimensional discrete wavelet transform or modified for use in a 2-dimensional inverse discrete wavelet transform. Moreover, the wavelet filter can be used in many different applications. For examples of the many different applications of wavelet analysis to signals, reference is made to a survey article entitled "Wavelet Analysis" by Bruce et. al. appearing in IEEE Spectrum, October 1996, pages 25 to 26. For a discussion of the different applications of wavelets in computer graphics, reference is made to "Wavelets for Computer Graphics", page 5, I. Stollinitz et. al. published 1996 by Morgan Kaufnann Publishers, Inc.--

Please amend the paragraph at page 4, lines 14-22, as follows:

--As mentioned above, the embodiments of the invention can be used in a discrete wavelet filter transform as well as an inverse discrete wavelet transform. For the

sake of simplicity, the term wavelet transform used herein or variations thereof[[,]] is taken to include both a forward and/or an inverse wavelet transform, unless the contrary intention appears. Similarly, the term pixel as used herein or variations thereof[[,]] is taken ~~to taken~~ to refer to the original pixels or wavelet coefficients of a digital image, unless the contrary intention appears. Similarly, the term digital image or variations thereof is taken to include an original image or a sub-band of that original image, or one or more associated sub-bands of wavelet transformed coefficients of that original image.--

Please amend the paragraph from page 4, line 28, to page 5, line 4, as follows:

--The filter 100 processes one horizontal band of the digital image after another. The digital image comprises one or more such horizontal bands, with the horizontal bands comprising a plurality of said rows of said pixels. The filter 100 initially commences processing the uppermost band of the image, and then processes the next adjacent band of the image, and so on until the lowermost band of the image is processed. Within each band, the filter [[200]] 100 processes one column of pixels after another. The filter 100 first processes the left column of the band, and then the next adjacent column of the band, and so on until the right column of the band is reached. For ease of understanding, the pixels in each column of a band from the topmost pixel to the lowermost pixel are represented as $x_0, x_1, x_2, \cdots x_{K'-1}$.--

Please amend the paragraph at page 6, lines 16-20, as follows:

--The multiplier and adder units 110 and 102 also accept inputs from either the partial result registers 109 or external buffer 107, depending on the control on the multiplexer 105. Namely, the multiplier and adder units 110 and 102 accept input from the external buffer 107 when computing the first 8 lines and ~~accepts~~ accept input from the partial result registers 109 for the last 8 lines.--

Please amend the paragraph from page 6, line 30, to page 7, line 8, as follows:

--For example, in the case where the filter 100 is currently processing the pixels (x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, and x_{15}), the multiplier and adder units 110 and 102 accept as input the last pixel and coefficient x_7 and d_3 from the partial result registers 109, which were previously stored in the partial result register 109 during the previous computation of the pixels (x_0, x_1, x_2, x_3, x_4, x_5, x_6, and x_7) of the same column and band. In the case where the filter 100 is currently processing the pixels (x_0 to x_7), the multiplier and adder units 110 and 102 accept as input the last pixel and coefficient x_{15} and d_7 from the external buffer 107. These inputs x_{15} and d_7 were previously stored in the external buffer 107 during the previous computation of the pixels (x_8 to x_{15}) of the same column but in the last previous band. In the case where there are no previous pixels (e.g. topmost band), the boundary is symmetrically reflected.--

Please amend the paragraph at page 7, lines 9-16, as follows:

--Figure 2 illustrates the cascading of the wavelet filter of Fig. 1 in time. In cycle 0, for the first set of inputs the wavelet filter 100 either symmetrically reflects the pixels near the boundary or reads the required partial results from the external buffer 107. The filter 100 then stores the partial results into the partial results register 109, and outputs the low and high pass coefficients. In cycle 1, the lifting lattice is 'moved' to the next set of input lines, reads input from the partial results register 109, and outputs the next low and high pass coefficients. The partial results are stored in the partial results register 109 and the external buffer 107.--

Please amend the paragraph at page 7, lines 17-26, as follows:

--The first embodiment has been described with respect to a filter 100 having two groups of eight input lines 101 and eight two-input multiplexers 103-1 to 103-8. It would be apparent to a person skilled in the art[[,]] that the number of groups can be increased (e.g. N = 4 groups) with a corresponding modification of the eight multiplexers, (ie. eight N-input multiplexers). The external buffer 107 is a slow memory storage and requires at least a memory size equivalent to the size of an input pixel and a high pass coefficient times the number of columns. The filter 100 stores in external memory the last input pixel and the last high pass coefficient x_{15} and d_7 of each column of each band for further processing in the next band. On the other hand, the partial results register 109 is a fast local storage and requires only a memory size of an input pixel and high pass coefficient.--

Please amend the paragraph at page 8, lines 4-7, as follows:

--In an alternate embodiment, the multiplexers 103-1, 103-2, 103-3, 103-4, 103-5, 103-6, 103-7, and 103-8 may be dispensed with. In this case, the multiplexing is done by ~~pixel fetching~~ a pixel-fetching mechanism implicitly. For instance, the pixels are fed to the lifting lattice from a frame buffer by an address generator.--

Please amend the paragraph at page 8, lines 8-21, as follows:

--Turning now to Fig. 3, there is shown a method for use in wavelet filtering in one dimension in accordance with a second embodiment. The method comprises in part a sub-procedure 300 for use in the wavelet filtering of a digital image. As mentioned previously, the digital image comprises a plurality of pixels arranged in a plurality of columns in the vertical direction and a plurality of rows in the horizontal direction. The digital image comprises one or more horizontal bands, each of which comprise a plurality of ~~said~~ the rows of pixels. The method processes one horizontal band of the digital image after another. The method initially commences processing the uppermost band of the image, and then processes the next adjacent band of the image, and so on until the lowermost band of the image is processed. Within each band, the method processes one column of pixels after another. The method first processes the left column of the band, and then the next adjacent column of the band, and so on until the right column of the band is reached. For ease of understanding, the pixels in each column of a band from the topmost pixel to the lowermost pixel are represented as $x_0, x_1, x_2, \ldots x_{K'-1}$.--

Please amend the paragraph at page 9, lines 3-12, as follows:

--The sub-procedure commences at step 301 and proceeds to step 302 where a counter M is set to one. As mentioned previously, the sub-procedure processes the pixels within a column of a band. The sub-procedure continues to a loop 306, 308, 310, 312, 314, and 316, where groups of a number K adjacent pixels are retrieved and processed one after another. A first group of K adjacent pixels (nearest the top boundary) is processed during the first pass of the loop 304 to [[313]] 316, a next group of K adjacent pixels (adjacent the first group) is processed during the second pass of the loop 304 to [[313]] 316, and so on. The processing of one group of pixels after another is achieved by incrementing the counter M. The sub-procedure terminates once K' pixels have been processed, when M.K. = K'. Preferably, K = 4 and K'= 4 or 8.--

Please amend the paragraph at page 9, lines 13-18, as follows:

--After the counter M has been set to one, the sub-procedure proceeds to step 304. In step 304, the method reads the corresponding previous partial results from an external buffer. These previous partial results ~~having being~~ have been stored in the external buffer during the processing of the sub-procedure of the same column in the last previous band of pixels. In the case where there are no previous pixels (e.g. topmost band) the boundary is symmetrically reflected.--

Please amend the paragraph at page 9, lines 19-27, as follows:

--After step 304, the method proceeds to loop 304 to 316. This loop will be described with respect to the general case where M = M'. During step 306, the sub-

procedure reads the M'th group of K adjacent pixels and then computes the low and high pass coefficients in step 308 in accordance with the equations (1) and (2) described above. In computing these high and low pass coefficients the method takes as input the M'th group of K adjacent pixels. [[It]] The method also accepts input from either the external buffer or a local register which stores the last pixel and the last high pass coefficient. The step 308 outputs to other discrete wavelet transform or entropy encoding procedures (not shown in the figure).--

Please amend the paragraph from page 9, line 28, to page 10, line 5, as follows:

--For example, in the case where the sub-procedure is currently processing the pixels (x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, and x_{15}), the sub-procedure step 308 accepts as input the last pixel and high pass coefficient x_7 and d_3 from the partial result ~~registers~~ register, which were previously stored in the partial result register during the previous computation of the pixels (x_0, x_1, x_2, x_3, x_4, x_5, x_6, and x_7) of the same column and band. In the case where the sub-procedure is currently processing the pixels (x_0 to x_7), the sub-procedure step 308 accepts as input the last pixel and high pass coefficient from the external buffer. These latter inputs were previously stored in the external buffer during the previous computation of the last group of pixels of the same column but in the last previous band. In the case where there are no previous pixels (e.g. topmost band) the boundary is symmetrically reflected.--

Please amend the paragraph at page 10, lines 6-9, as follows:

--After step 308, the method proceeds to a decision block 310, where a check is made to determine whether M.K. = K'. If the decision block returns false, then the method proceeds to step 312, where the last pixel and last high pass coefficient are stored in the local partial results register for use in the computation 308 of the next group of pixels.--

Please amend the paragraph at page 10, lines 10-14, as follows:

--In the next step 314, the counter M is incremented and the last pixel and the last high pass coefficient are read in step 316. The method then returns to step 306 for the next pass of the loop 306 to 316. In the event[[,]] the decision block 310 returns true, then the last pixel and the last high pass coefficient are stored in the external buffer and the sub-procedure finishes for the current column of the band.--

Please amend the paragraph at page 10, lines 17-20, as follows:

--The aforementioned preferred method comprise a particular control flow. There are many other variants of the preferred method which use different control flows without departing the spirit or scope of the invention. Furthermore one or more of the steps of the preferred method may be performed in parallel rather ~~sequential~~ than sequentially.--

Please amend the paragraph from page 12, line 20, to page 13, line 2, as follows:

--In this modified embodiment, the 9/7 forward transform wavelet filter computes the low pass coefficients and high pass coefficients in accordance with the following formulae:

$$d'_n = x_{2n+1} + \alpha(x_{2n} + x_{2n-2}) \qquad n = 0,1,\ldots,K-1 \qquad (14)$$

$$s'_n = x_{2n} + \beta(d'_{n-1} + d'_n) \qquad n = 0,1,\ldots,K-1 \qquad (15)$$

$$d_n = d'_n + \gamma(s'_n + s'_{n+1}) \qquad n = -1,0,\ldots,K-2 \qquad (16)$$

$$sn = s'_n + \delta(d_{n-1} + d_n) \qquad n = -1,0,\ldots,K-2 \qquad (17)$$

where d_n are all the high pass coefficients, s_n are all the low pass coefficients, d'_n and s'_{n-1} are intermediate values, $\alpha = -1.5861$, $\beta = -0.052980$, $\gamma = 0.88291$, and $\delta = 0.44351$. The multiplier and adder units of Fig. 1 will be modified accordingly to compute the coefficients in accordance with equations (14) to (17). This filter takes as input pixels x_1 to x_k and also requires as input the pixel x_0, the intermediate values d'_{n-1} and s'_{n-1} and the high pass coefficient d_{-2} from the previous band, which are stored in the partial register 109 or external buffer 107.--